Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application.

Listing of Claims:

1. **(Currently Amended)** A method of automatically determining an allowable order of changes in a distributed system, the method comprising the steps of:

receiving a request for change;

wherein the request for change describes a task to be done on at least one target system, and a deadline by which a change needs to be completed;

determining existing relationship descriptions between components of the distributed system, wherein the components of the distributed system are implemented on a plurality of managed resources;

transforming acquired relationships into ordered tasks that are linked by temporal ordering constraints which describe when tasks can begin in relation to one another, wherein the temporal ordering constraints are selected from the group consisting of: Finish-to-Start, Start-to-Start, Finish-to-Finish, and Start-to-Finish; and

creating an order of changes taking into account ~~task relationship~~ the temporal ordering constraints, wherein creating the order of changes includes determining whether the ordered changes are conflicting and flagging such conflicts and further includes an estimate of the time required to complete a change;

wherein the order of changes transitions the target system from one workable state into another workable state.

2. **(Original)** The method of Claim 1, wherein the order of changes is sequential.

3. **(Original)** The method of Claim 1, wherein the order of changes is concurrent.

4. **(Previously Presented)** The method of Claim 1, further comprising refining an incoming request for change by breaking the incoming request down into sub-requests.

5. **(Previously Presented)** The method of Claim 4, further comprising computing an allowable order of changes by interacting with the distributed system.

6. **(Cancelled)** The method of Claim 1, wherein creating the order of changes includes determining whether the ordered changes are conflicting and flagging such conflicts.

7. **(Previously Presented)** The method of Claim 1, wherein the ordered changes are partially ordered.

8. **(Previously Presented)** The method of claim 1, wherein the ordered changes are totally ordered.

9. **(Cancelled)** The method of Claim 1, wherein the order of changes includes an estimate of the time required to complete a change.

10. **(Previously Presented)** The method of Claim 4, wherein a total change time is minimized by exploiting parallelism between change tasks.

11. (**Original**) The method of Claim 1, wherein the creation of the order of changes further takes into account a requested change management operation.

12. (**Original**) The method of Claim 1, wherein a requester identifies one or more target systems within the distributed system by name.

13. (**Original**) The method of Claim 12, wherein the names of the target systems are unique physical identifiers.

14. (**Original**) The method of Claim 12, wherein the names of the target systems are logical names which refer to one or more physical systems.

15. (**Original**) The method of Claim 1, wherein a requester does not identify one or more target systems within the distributed system by name.

16. (**Original**) The method of Claim 1, further comprising the steps of accessing and evaluating policy rules representing best practices.

17. (**Original**) The method of Claim 16, wherein the best practices include updating all affected software artifacts when a software artifact is updated.

18. (**Original**) The method of Claim 16, wherein the best practices include having a given set of software components installed on different systems.

19. (**Original**) The method of Claim 1, wherein one or more of the order of changes are persistently stored after being created.

20. (**Previously Presented**) The method of Claim 1, wherein a component is one of a service, an application, middleware, hardware, an operating system, a storage system, a network device, and a system associated with a computing environment.

21. (**Currently Amended**) A system for automatically determining an allowable order of changes in a distributed system, the system comprising:

a processor; and

a memory storing code accessible by the processor to:

determine existing relationship descriptions between components of the distributed system, wherein the components of the distributed system are implemented on a plurality of managed resources;

transform acquired relationships into ordered tasks that are linked by temporal ordering constraints which describe when tasks can begin in relation to one another, wherein the temporal ordering constraints are selected from the group consisting of: Finish-to-Start, Start-to-Start, Finish-to-Finish, and Start-to-Finish; and

create an order of changes taking into account ~~task relationship~~ the temporal ordering constraints, wherein creating the order of changes includes determining whether the ordered changes are conflicting and flagging such conflicts and further includes an estimate of the time required to complete a change;

wherein the order of changes transitions the target system from one workable state into another workable state.

22. **(Original)** The system of Claim 21, wherein the order of changes is sequential.

23. **(Original)** The system of Claim 21, wherein the order of changes is concurrent.

24. **(Previously Presented)** The system of Claim 21, further comprising an arrangement for refining an incoming request for change by breaking the incoming request down into sub-requests.

25. **(Previously Presented)** The system of Claim 24, further comprising an arrangement for computing an allowable order of changes by interacting with the distributed system.

26. **(Cancelled)** The system of Claim 21, wherein creating the order of changes includes determining whether the ordered changes are conflicting and flagging such conflicts.

27. **(Previously Presented)** The system of Claim 21, wherein the ordered changes are partially ordered.

28. **(Previously Presented)** The system of claim 21, wherein the ordered changes are totally ordered.

29. **(Cancelled)** The system of Claim 21, wherein the order of changes includes an estimate of the time required to complete a change.

30. **(Previously Presented)** The system of Claim 24, wherein a total change time is minimized by exploiting parallelism between change tasks.

31. (**Original**) The system of Claim 21, wherein the creation of the order of changes further takes into account a requested change management operation.

32. (**Original**) The system of Claim 21, wherein a requester identifies one or more target systems within the distributed system by name.

33. (**Original**) The system of Claim 32, wherein the names of the target systems are unique physical identifiers.

34. (**Original**) The system of Claim 32, wherein the names of the target systems are logical names which refer to one or more physical systems.

35. (**Original**) The system of Claim 21, wherein a requester does not identify one or more target systems within the distributed system by name.

36. (**Original**) The system of Claim 21, further comprising an arrangement for accessing and evaluating policy rules representing best practices.

37. (**Original**) The system of Claim 36, wherein the best practices include updating all affected software artifacts when a software artifact is updated.

38. (**Original**) The system of Claim 36, wherein the best practices include having a given set of software components installed on different systems.

39. (**Original**) The system of Claim 21, wherein one or more of the order of changes are persistently stored after being created.

40. **(Previously Presented)** The system of Claim 21, wherein a component is one of a service, an application, middleware, hardware, an operating system, a storage system, a network device, and a system associated with a computing environment.

41. **(Currently Amended)** A program storage device readable by machine, tangibly embodying a program of instructions executable by the machine to perform method steps for automatically determining an allowable order of changes in a distributed system, said method comprising the steps of:

receiving a request for change;

wherein the request for change describes a task to be done on at least one target system, and a deadline by which a change needs to be completed;

determining existing relationship descriptions between components of the distributed system, wherein the components of the distributed system are implemented on a plurality of managed resources;

transforming acquired relationships into ordered tasks that are linked by temporal ordering constraints which describe when tasks can begin in relation to one another, wherein the temporal ordering constraints are selected from the group consisting of: Finish-to-Start, Start-to-Start, Finish-to-Finish, and Start-to-Finish; and

creating an order of changes taking into account ~~task relationship~~ the temporal ordering constraints, wherein creating the order of changes includes determining

whether the ordered changes are conflicting and flagging such conflicts and further includes an estimate of the time required to complete a change;

wherein the order of changes transitions the target system from one workable state into another workable state..

42. (**New**) The method of Claim 1, further comprising:

building a Task Graph which represents each task to be completed within an overall job;

constructing an Annotated Task Graph by assigning estimated durations to each task within the Task Graph; and

computing a makespan for the overall job represented by the Task Graph, wherein the Annotated Task Graph is returned to an administrator.

43. (**New**) The system of Claim 21, further comprising an arrangement for:

building a Task Graph which represents each task to be completed within an overall job;

constructing an Annotated Task Graph by assigning estimated durations to each task within the Task Graph; and

computing a makespan for the overall job represented by the Task Graph, wherein the Annotated Task Graph is returned to an administrator.

44. (**New**) The program storage device of Claim 41, which further:

builds a Task Graph which represents each task to be completed within an overall job;

constructs an Annotated Task Graph by assigning estimated durations to each task within the Task Graph; and

computes a makespan for the overall job represented by the Task Graph, wherein the Annotated Task Graph is returned to an administrator.